UNIVERSITÉ DE MONTRÉAL

CENTRE DE RECHERCHE SUR LES TRANSPORTS

CASE POSTALE 6128, STATION A, MONTRÉAL, QUÉBEC, H3C 3J7

RECHERCHE D'UN ALGORITHME EFFICACE POUR LE PROBLEME DU CHOIX OPTIMAL D'UN RESEAU

par

René Dionne

Cette recherche appliquée a été subventionnée en partie par "Ford Motor Company of Canada".

Université de Montréal
Publication # 174 - Département d'informatique
Publication # 7 - Centre de Recherche sur les Transports
Mai 1974

1. Introduction

Dans un algorithme proposé par Hoang (1971) pour résoudre le problème du choix optimal d'un réseau, on retrouve souvent la situation suivante: connaissant la matrice des longueurs (des plus courtes chaînes) d'un réseau symétrique et modifiant la longueur d'une seule arête du réseau, il faut calculer la nouvelle matrice des longueurs. Or les algorithmes que nous avons mis au point dans un travail précédent (Dionne (1973)) sont tout-à-fait adaptés à ce genre de calcul. Nous avons donc voulu les implanter dans l'algorithme de Hoang (1971), dans le but de réduire le temps de calcul. En cours de travail, nous avons apporté de nouvelles idées, qui allaient permettre d'obtenir un algorithme encore plus efficace.

Dans le présent rapport, nous commençons par présenter l'algorithme initial (celui que Hoang (1971) appelle "branch search"). Nous le faisons dans une notation et selon un point de vue qui diffèrent de ceux de Hoang, mais qui nous paraissent plus appropriés pour expliquer les modifications que nous ferons subir à l'algorithme initial par la suite. Nous décrivons ensuite comment nous avons procédé, par modifications successives de l'algorithme initial fondées sur des résultats numériques, pour améliorer l'algorithme initial. L'algorithme résultant diffère sensiblement de l'algorithme initial et il se révèle beaucoup plus efficace.

2. Présentation de l'algorithme initial

2.1 Notation*

Soit $R = (G,L)$ un réseau symétrique connexe défini par la donnée d'un graphe $G = (N,A)$ non orienté et connexe ainsi que d'une fonction L, définie sur A, qui associe à chaque arête de G une longueur positive.

$G = (N,A)$

$N = \{1,2,\ldots,n\}$ = ensemble des points ou sommets du graphe G

$A = \{a_1,a_2,\ldots,a_m\}$ = ensemble des arêtes du graphe G

$M = \{1,2,\ldots,m\}$

$L : A \longrightarrow \mathbb{R}^+$

$a_j \xmapsto{L} \ell_j,\ \ell_j > 0,\ j \in M$.

Nous employons G_p comme désignation générique d'un graphe partiel de G :

$$G_p = (N,A') \quad , \quad A' \subseteq A .$$

Un réseau partiel R_p se trouve défini par la donnée de G_p et par la restriction à A' de la fonction L :

$$R_p = (G_p,\ L \text{ restreinte à } A') .$$

R_p est dit <u>connexe</u> si G_p est connexe.

---

* Nous suivons la terminologie établie par Berge (1967). Ainsi, nous disons "chaîne" et non pas "chemin", dans le cas d'un réseau non orienté.

Une correspondance biunivoque s'établit naturellement entre l'ensemble des réseaux partiels (R_p) de R et l'ensemble des vecteurs caractéristiques (X^p) des A', $A' \subseteq A$:

$$R_p \longleftrightarrow X^p$$

$$X^p = (x_1^p, x_2^p, \ldots, x_m^p) \quad , \quad x_j^p = \begin{cases} 1, & \text{si } a_j \in A' \\ 0, & \text{sinon .} \end{cases}$$

Posons, par définition:

$c_{ij}(X^p)$ = longueur de la plus courte chaîne du sommet i au sommet j dans R_p $(i,j \in N)$

$$C(X^p) = \left(c_{ij}(X^p) \right)_{n \times n}$$

$$F(X^p) = \sum_{\substack{i,j \\ i<j}} c_{ij}(X^p) \quad .$$

Si l'arête a_k fait partie du réseau* X^p et a pour extrémités les sommets i et j, alors a_k peut s'écrire $a_k = (i,j)$ et nous

*Note: Puisque R_p et X^p se correspondent biunivoquement, nous dirons "le réseau X^p" plutôt que "le réseau partiel représenté par le vecteur X^p".

définissons:

$$f_k(X^p) = c_{ij}(X^p - e_k) - c_{ij}(X^p)$$

$$Q_k(X^p) = F(X^p - e_k) - F(X^p)$$

$$Q_k(X^p) \geq f_k(X^p)$$

$$F(X^p - e_k) = F(X^p) + Q_k(X^p) \geq F(X^p) + f_k(X^p)$$

où e_k est le $k^{ième}$ vecteur canonique à m composantes, i.e. $e_k = (0,0,\ldots,0,1,0,0,\ldots,0)$

un seul 1 *en position* k .

$F(X^p)$ constitue une mesure de la "longueur totale" du réseau X^p. $f_k(X^p)$ correspond à l'augmentation de longueur que l'on fait subir à la plus courte chaîne entre i et j lorsqu'on enlève l'arête a_k du réseau X^p ; f_k n'est donc qu'une mesure locale de l'effet résultant de la suppression de l'arête a_k. Quant à $Q_k(X^p)$, il indique l'augmentation totale de longueur dans le réseau X^p causée par la suppression de a_k ; Q_k correspond donc à une mesure globale de l'effet résultant de cette suppression. $Q_k(X^p)$ et $f_k(X^p)$ constituent donc des mesures de l'utilité de l'arête a_k dans le réseau X^p : plus $Q_k(X^p)$ — ou $f_k(X^p)$ — est grand, plus il est utile de conserver l'arête a_k dans le réseau X^p. Evidemment, Q_k est une meilleure mesure d'utilité que f_k.

Dans le présent rapport, nous appelons <u>solution</u> n'importe quel vecteur à m composantes dont toutes les composantes sont 0 ou 1. A

toute solution correspond un réseau partiel et vice-versa. L'ensemble de toutes les solutions s'écrit

$$S = \{X = (x_1, x_2, \ldots, x_m) \mid \quad x_j = 0 \text{ ou } 1,\ j \in M\} .$$

Nous appelons solution réalisable une solution qui satisfait les contraintes du problème du choix optimal d'un réseau (défini ci-dessous, cf. 2.2), et nous appelons solution optimale une solution réalisable qui minimise la fonction économique du même problème.

2.2 Définition du problème du choix optimal d'un réseau

Le problème à résoudre s'énonce comme suit:

Etant donné un réseau R et un budget B, trouver un réseau partiel R_p tel que:

- R_p est connexe (contrainte 1)
- $\sum_{j=1}^{m} \ell_j x_j^p \leq B$ (contrainte 2)
- $F(X^p)$ est minimum.

On saisit immédiatement la nature combinatoire du problème lorsqu'on songe que l'évaluation de la fonction économique F exige la connaissance de la longueur de $n(n-1)/2$ plus courtes chaînes, que la vérification de la contrainte 1 peut exiger autant d'efforts que l'évaluation de la fonction économique et que l'ensemble S de toutes les solutions est de cardinalité 2^m.

2.3 L'arborescence associée

L'algorithme de Hoang (1971) est une P.S.E., une Procédure d'exploration par Séparation et Evaluation (suivant la terminologie établie par B. Roy (1969); en anglais, branch-and-bound method, cf. Lawler and Wood (1966)). Grosso modo, une P.S.E. cherche à séparer l'ensemble S de toutes les solutions en sous-ensembles de cardinalité de plus en plus petite, donnant lieu à des problèmes de plus en plus faciles à résoudre, la minimisation se faisant sur chacun de ces sous-ensembles plutôt que sur l'ensemble S . A partir de S , une P.S.E. génère donc, à l'aide d'une partition, un certain nombre de sous-ensembles; chacun des sous-ensembles ainsi générés sera évalué; cette EVALUATION cherche à éliminer chaque sous-ensemble généré, ou bien parce qu'il ne renferme aucune solution intéressante, ou bien parce que la meilleure solution réalisable qu'il contient surclasse (et remplace) la meilleure solution réalisable obtenue jusqu'alors; tout sous-ensemble généré et non éliminé sera, à son tour, séparé par une partition (SEPARATION). Et ainsi de suite jusqu'à ce que tous les sous-ensembles générés aient été éliminés. La meilleure solution réalisable conservée se trouve alors être la solution optimale.

La relation de descendance qui lie un sous-ensemble à ceux qu'il engendre par séparation se représente commodément par une arborescence, que nous appellerons arborescence associée (pour un exemple, voir fig. 1).

Dans l'algorithme initial (Hoang (1971)), la séparation d'un sous-ensemble de S s'effectue à l'aide d'une bi-partition. Les sous-ensembles considérés sont du type suivant:

$$E = \left\{ (x_1, x_2, \ldots, x_m) \;\middle|\; \begin{array}{l} x_j \text{ peut prendre n'importe quelle valeur (0 ou 1)} \\ \quad \text{si } j \in L \\ x_j \text{ a une valeur fixe si } j \in M - L \end{array} \right\}$$

où L est l'ensemble des indices des variables libres

et $M - L$, l'ensemble des indices des variables fixées.

$|E| = 2^{|L|}$.

Les bi-partitions considérées sont du type suivant: étant donné un sous-ensemble E du type décrit ci-dessus, on choisit un indice $r \in L$ et on sépare E en ses deux sous-ensembles,

$$E \cap \{(x_1, x_2, \ldots, x_m) \in S \mid x_r = 0\} \quad \text{et} \quad E \cap \{(x_1, x_2, \ldots, x_m) \in S \mid x_r = 1\} .$$

Nous appelons <u>variable séparatrice</u> la variable x_r qui définit la séparation.

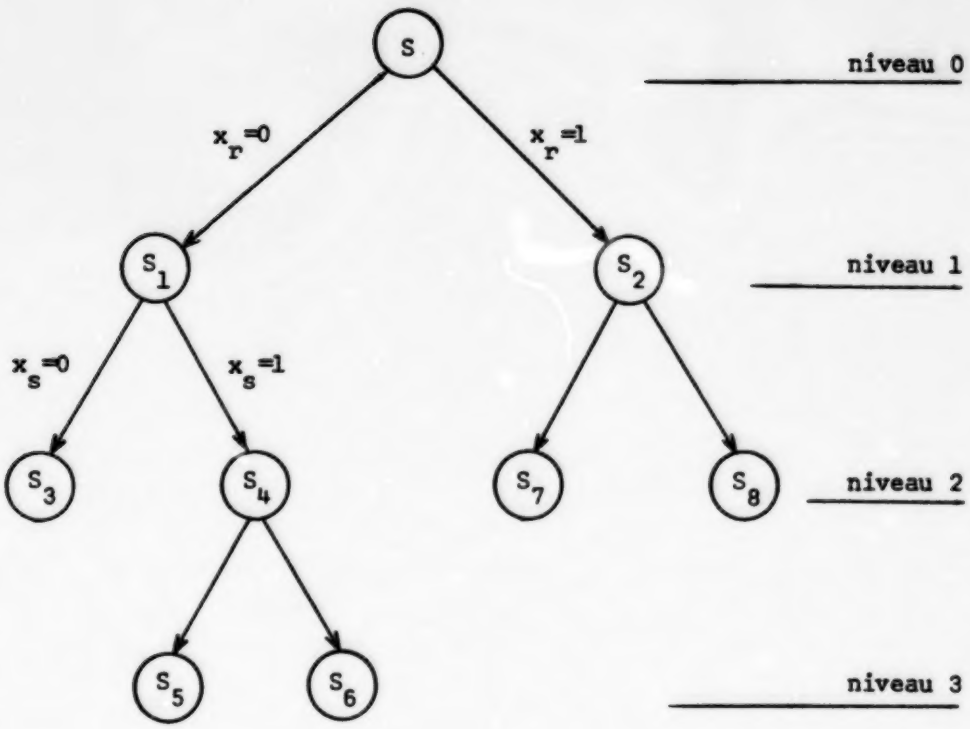


Figure 1

Exemple d'arborescence générée par l'algorithme:

L'ensemble S se trouve associé au sommet racine (à ce moment, L = M) . Puis S se trouve séparé en 2 sous-ensembles S_1 et S_2 par le choix d'un indice $r \in L$.

$S_1 = \{(x_1,x_2,\ldots,x_m) \in S \mid x_r = 0\}$ et $S_2 = \{(x_1,x_2,\ldots,x_m) \in S \mid x_r = 1\}$.

L devient $M - \{r\}$.

Par le choix d'un indice $s \in L$, x_s devient la variable séparatrice pour S_1, qu'on sépare en S_3 et S_4 .

$S_3 = \{(x_1,x_2,\ldots,x_m) \in S \mid x_r = 0, x_s = 0\}$

et $S_4 = \{(x_1,x_2,\ldots,x_m) \in S \mid x_r = 0, x_s = 1\}$.

Etc.

L'algorithme initial génère et évalue les sous-ensembles de façon séquentielle (dans la notation de Roy (1969), il s'agit d'une P.S.E.S. — S pour séquentielle). Lorsqu'on évalue un sous-ensemble E, il importe de connaître quelles sont les variables fixées dans E (avec leur valeur) ainsi que l'ordre dans lequel elles ont été fixées. A cette fin, nous utilisons une pile P définie comme suit.

Lorsqu'on évalue un sous-ensemble E, les éléments dans la pile sont les indices (précédés ou non du signe " - ") des variables fixées dans E :

$$\left|\begin{array}{ll} x_j = 0 & \text{est représenté par } -j, \\ x_j = 1 & \text{est représenté par } j\ . \end{array}\right.$$

L'indice au bas de la pile est celui de la première variable fixée et l'indice sur le dessus de la pile, celui de la dernière variable fixée. L'exemple suivant illustrera cette notation:

$P =$ | 4 | signifie qu'on a fixé d'abord $x_5 = 0$
| -7 | puis $x_3 = 1$
| 3 | puis $x_7 = 0$
| -5 | et enfin $x_4 = 1$.

Ainsi, la pile P décrit complètement le sous-ensemble qu'on est en train d'évaluer. Nous noterons ce sous-ensemble par E^P .

$$E^P = \left\{ (x_1, x_2, \ldots, x_m) \in S \;\middle|\; \begin{array}{l} x_j = 0 \quad \text{si} \quad -j \quad \text{apparaît dans } P \\ \\ x_j = 1 \quad \text{si} \quad j \quad \text{apparaît dans } P \end{array} \right\}$$

Quant à l'ensemble des variables libres dans E^P, nous le désigne-

rons par L^P :

L^P = ensemble des indices des variables dont l'indice n'apparaît pas dans la pile P .

Dans le sous-ensemble E^P, nous distinguons la solution qui représente le réseau comptant le plus grand nombre d'arêtes et nous la notons X^P . Dorénavant, X^P ne désignera plus une solution quelconque, mais un élément bien particulier de E^P .

$$X^P = (x_1^P, x_2^P, \ldots, x_m^P) \quad , \quad \text{où} \quad x_j^P = \begin{cases} 0 \text{ , si } -j \text{ apparaît dans } P \\ 1 \text{ , autrement.} \end{cases}$$

$\forall X \in E^P$, on a : $F(X^P) \leq F(X)$.

2.4 La borne inférieure I^P

Lorsqu'on examine le sous-ensemble E^P , on cherche à l'éliminer de diverses façons. L'une de ces façons consiste à calculer une borne inférieure de l'ensemble $\{F(X) \mid X \in E^P$ et X réalisable$\}$, dénotée par I^P . Si $I^P > F(X^o)$, où X^o représente la meilleure solution courante, on dit que la borne effectue une coupe (ou tout simplement coupe) et on peut alors éliminer le sous-ensemble E^P , car il ne contient sûrement aucune solution réalisable X meilleure que X^o (i.e. telle que $F(X) < F(X^o)$) . Voyons comment obtenir cette borne.

Théorème I: $\forall X \in E^P$, $X = (x_1, x_2, \ldots, x_m)$, on a:

$$F(X) \geq F(X^P) + \sum_{j \in L^P} \bar{x}_j \, f_j(X^P) \, ,$$

où $\bar{x}_j = 1 - x_j$.

Le théorème est de Hoang (1971) et nous offrons la preuve suivante.

Preuve: Soit $X \in E^P$.

Si le réseau X est disconnexe, alors $F(X) = \infty$ et le théorème est nécessairement vérifié. Supposons donc X connexe.

$X \in E^P$ signifie que X est obtenu de X^P par suppression d'un certain nombre d'arêtes, disons s ($s \leq m$) . Pour simplifier, on supposera que les arêtes sont numérotées de façon que les s arêtes supprimées soient les s premières.

$$(*) \quad \begin{cases} x_j = 0 \text{ , mais } x_j^P = 1 \text{ , pour } j \leq s \\ x_j = x_j^P \text{ , pour } j > s \\ \underline{\text{Note}}: L^P \supseteq \{1,2,\ldots,s\} \text{ , mais } \forall j \in L^P \ni j > s, \text{ on a: } x_j = 1 . \end{cases}$$

Supposons que le réseau Y diffère du réseau X par le seul fait qu'il contient une arête de plus que X , disons l'arête a_s .

$$(**) \quad X = Y - e_s \quad \text{i.e.} \begin{cases} y_s = 1, \text{ mais } x_s = 0 , \\ y_j = x_j , \ \forall j \in M - \{s\} . \end{cases}$$

(***) $F(X) \geq F(Y) + f_s(Y)$, ce qui découle de la définition de $f_s(Y)$.

(*En effet, si* $a_s = (u,v)$,
alors $c_{ij}(X) \geq c_{ij}(Y)$, $ij \neq uv$,
et par définition, $c_{uv}(X) = c_{uv}(Y) + f_s(Y)$.)

La preuve procède par induction sur s .

Si $s = 1$, $F(X) \geq F(X^P) + f_s(X^P)$, par (***) .

Supposons le théorème vrai pour s-1 et démontrons-le pour s quelconque.

$$F(X^p) + \sum_{j \in L^p} \bar{x}_j \, f_j(X^p) = F(X^p) + \sum_{j=1}^{s-1} \bar{x}_j \, f_j(X^p) + f_s(X^p) \text{ , par (*)}$$

$$= F(X^p) + \sum_{j=1}^{s-1} \bar{y}_j \, f_j(X^p) + f_s(X^p) \text{ , par (**)}$$

$$= F(X^p) + \sum_{j \in L^p} \bar{y}_j \, f_j(X^p) + f_s(X^p) \text{ ,}$$

par (*, note) et par (**)

$$\leq F(Y) + f_s(X^p) \text{ , par l'hypothèse d'induction}$$

$$< F(Y) + f_s(Y) \text{ , parce que } f_s(Y) > f_s(X^p)$$

(en effet, lorsqu'on supprime l'arête $a_s = (u,v)$, ou bien la longueur de la plus courte chaîne de u à v ne change pas ($f_s = 0$), ou bien la plus courte chaîne de u à v est remplacée par une chaîne plus longue. Dans ce dernier cas, les possibilités de remplacement dans Y ne constituent qu'une partie des possibilités de remplacement dans X^p, et donc $f_s(Y) > f_s(X^p)$.)

$$\leq F(X) \text{ , par (***) .} \qquad \text{C.Q.F.D.}$$

Considérons que $X^+ = (x_1^+, x_2^+, \ldots x_m^+)$ est la solution optimale du problème (1):

$$(1)\begin{cases} \text{MIN} \quad F(X^P) + \sum\limits_{j \in L^P} \bar{x}_j \, f_j(X^P) \\ \text{en respectant les contraintes} \\ \quad (a) \quad X = (x_1, x_2, \ldots, x_m) \text{ représente un réseau connexe} \\ \quad (b) \quad \sum\limits_{j=1}^{m} \ell_j \, x_j \leq B \\ \quad (c) \quad X \in E^P \ . \end{cases}$$

Soit $\alpha = F(X^P) + \sum\limits_{j \in L^P} \bar{x}_j^+ \, f_j(X^P)$. Alors, par le théorème I, α vérifie l'inégalité $\alpha \leq \min \{F(X) \mid X \in E^P \text{ et } X \text{ réalisable}\}$ et constitue une borne inférieure de l'ensemble

$$\{F(X) \mid X \in E^P \text{ et } X \text{ réalisable}\} \ .$$

Cependant, résoudre le problème (1) s'avère coûteux en temps de calcul et d'ailleurs nullement avantageux, puisque cela exige un effort de calcul du même ordre que la recherche pure et simple de la meilleure solution réalisable dans E^P. On se contentera plutôt de calculer une borne inférieure un peu plus faible, mais tout de même utile, en relâchant les contraintes (a) et (c) du problème (1), ce qui donne lieu au problème (2).

$$
(2)\quad \begin{cases}
\text{MIN} \quad F(X^P) + \sum\limits_{j \in L^P} \bar{x}_j \, f_j(X^P) \\
\text{en respectant les contraintes} \\
\text{(b)} \quad \sum\limits_{j=1}^{m} \ell_j \, x_j \leq B \\
\text{(c')} \; x_j = 0 \quad \text{si} \quad -j \quad \text{apparaît dans la pile} \quad P \\
\qquad x_j = 1 \quad \text{si} \quad j \quad \text{apparaît dans la pile} \quad P \\
\qquad 0 \leq x_j \leq 1 \,, \; x_j \text{ réel, si } j \in L^P \,.
\end{cases}
$$

On constate que le problème (2) est tout simplement un problème de sac alpin (knapsack problem) dans lequel certaines variables peuvent prendre des valeurs fractionnaires. Ce problème s'avère très facile à résoudre. En effet, supposons que les variables ont été ordonnées (dans les calculs, on modifiera les indices) de telle façon que

$$L^P = \{1,2,\ldots,e\} \quad , \quad \text{où} \quad e \leq m \,,$$

$$\text{et que} \quad \frac{f_1(X^P)}{\ell_1} \leq \frac{f_2(X^P)}{\ell_2} \leq \ldots \leq \frac{f_e(X^P)}{\ell_e} \quad ,$$

alors la solution optimale évidente de (2) s'écrit (cf. Dantzig (1957)):

$$X^* = (x_1^*, x_2^*, \ldots, x_m^*)$$

$$\text{où} \begin{cases} x_j^* = 0 \quad , \text{ si } j < r \\ x_j^* = 1 \quad , \text{ si } j > r \\ x_r^* = 1 - \left[\left(b - \sum_{i=1}^{r-1} \ell_i \right) / \ell_r \right] \quad , \\ \text{où } r \text{ est le plus petit entier } \ni\ 0 < r \leq e \text{ et } \sum_{i=1}^{r} \ell_i > b \ . \end{cases}$$

$$\boxed{I^P = F(X^P) + \sum_{j \in L^P} \bar{x}_j^* \, f_j(X^P) \ .}$$

2.5 Le critère de choix de la variable séparatrice

Hoang (1971) propose que l'indice r qui définit la variable séparatrice x_r soit justement le même indice r qu'on calcule en solutionnant le problème (2), c'est-à-dire le plus petit entier $\ni$

$$0 < r \leq e \quad \text{et} \quad \sum_{i=1}^{r} \ell_i > B \ .$$

Comme nous serons appelés plus loin à modifier ce critère de choix de la variable séparatrice, nous introduisons dès maintenant une nouvelle notation qui permettra d'écrire d'une façon concise n'importe quel critère de choix de la variable séparatrice: dans cette notation, nous écrivons entre paranthèses 1) la mesure d'utilité par rapport à laquelle on ordonne les variables (par exemple, f_j/ℓ_j) , et 2) le rang de la va-

riable à choisir comme variable séparatrice. Dans cette notation, le critère de choix proposé par Hoang (1971) s'écrit $(f_j/\ell_j, r)$.

2.6 La solution initiale

Hoang (1971) propose deux méthodes pour obtenir une solution initiale (réalisable), mais sans mentionner quels sont les avantages respectifs de ces méthodes.

La première méthode, que nous appellerons méthode du graphe partiel connexe minimal (minimum spanning tree) est la plus simple. Elle consiste à obtenir le graphe partiel connexe dont la somme des longueurs des arêtes est minimale et à lui ajouter une à une les arêtes les plus courtes parmi celles qui n'ont pas encore été choisies aussi longtemps qu'on ne dépasse pas le budget.

La deuxième méthode utilise les Q_j et nous l'appellerons méthode des Q_j . Elle exige plus de calculs, mais devrait fournir une meilleure solution initiale. Elle procède comme suit:

(i) Poser $X = (1,1,\ldots,1)$.

(ii) $\forall j \in M \ni x_j = 1$, évaluer $Q_j(X)$. Puis

a) si $Q_j(X) = \infty \ \forall_j \ni x_j = 1$, recommencer les calculs en employant la méthode du graphe partiel connexe minimal;

b) sinon, trouver $k \ni Q_k(X) = \min_{\{j \in M | x_j = 1\}} Q_j(X)$

et poser $x_k = 0$. Aller en (iii).

(iii) Si $\sum_{j=1}^{m} \ell_j x_j > B$, aller en (ii). Sinon, aller en (iv).

(iv) Si le budget le permet, tenter de redonner à une composante nulle de X la valeur 1 de façon à minimiser $F(X)$. Répéter jusqu'à ce qu'il ne soit plus possible de modifier aucune composante de X . X est alors la solution initiale cherchée.

2.7 Définition de l'algorithme de Hoang

INITIALISATION: On détermine d'abord une solution initiale X^o , par l'une des méthodes précitées. On pose alors $E^p = S$, c'est-à-dire que la pile P est vide, et on procède à l'évaluation de E^p .

EVALUATION: On examine le sous-ensemble E^p pour tenter de l'éliminer.

(1) On vérifie si E^p contient au moins un réseau connexe; cela revient à se demander si X^p est un réseau connexe. Si tel n'est pas le cas, E^p se trouve éliminé et on effectue un retour en arrière (voir détails ci-après).

(2) Si X^p est connexe (contrainte 1), on vérifie si X^p satisfait la contrainte (2) du problème du choix optimal d'un réseau. Si oui, X^p est une solution réalisable et c'est, de toutes les solutions dans E^p , celle qui donne la plus petite valeur à la fonction économique F , puisque c'est elle qui comporte le plus d'arêtes. Dans ce cas, si

$(F(X^p) , \sum_{j=1}^{m} \ell_j x_j^p)$ est lexicographiquement plus petit que $(F(X^o) , \sum_{j=1}^{m} \ell_j x_j^o)$, alors X^p remplace X^o comme meilleure

solution réalisable courante. E^p se trouve ainsi éliminé et on fait un retour en arrière.

(3) Si X^p ne respecte pas la contrainte 2, on tente à nouveau d'éliminer E^p en calculant la borne inférieure I^p.
Si $I^p > F(X^o)$, on élimine E^p et on effectue un retour en arrière; sinon, on procède à la séparation de E^p.

SEPARATION: Le calcul de I^p a permis de déterminer un indice $r \in L^p$. On sépare E^p en 2 sous-ensembles:

$$E_o = E^p \cap \{(x_1,x_2,\ldots,x_m) \in S \mid x_r = 0\}$$

et $E^p \cap \{(x_1,x_2,\ldots,x_m) \in S \mid x_r = 1\}$.

On ajoute alors $-r$ sur le dessus de la pile P. E^p se trouve ainsi remplacé par E_o. On procède alors à l'évaluation de E^p.

RETOUR EN ARRIERE (backtracking): Effectuer un retour en arrière consiste à revenir à l'évaluation du dernier sous-ensemble généré mais non encore évalué. Lorsque tous les sous-ensembles générés ont été examinés, l'algorithme s'arrête et X^o est la solution optimale. ◆Fin.

L'algorithme n'élimine que les sous-ensembles qu'on sait ne sûrement pas contenir la solution optimale. En ce sens, il effectue une énumération partielle mais exhaustive des solutions qui conduit à la solution optimale. De plus, l'algorithme s'arrête après un nombre fini d'opérations, puisque le nombre de sous-ensembles de S est fini.

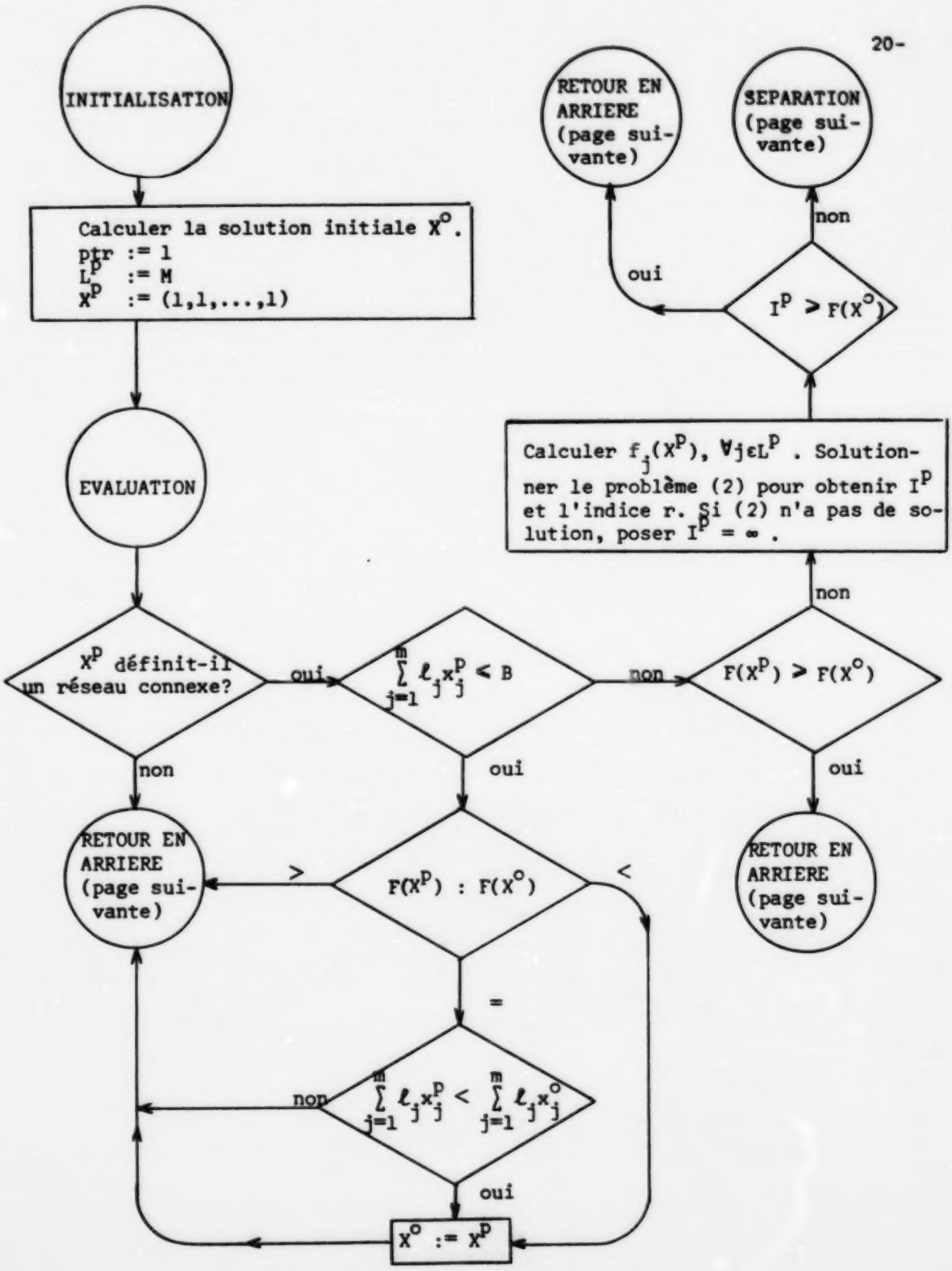
Nous allons maintenant donner l'organigramme de l'algorithme, ce qui constituera une définition formelle de l'algorithme. Dans cet organigramme, la pile P est représentée par un vecteur P à m composantes

ainsi que par un pointeur ptr , qui indique toujours le sommet de la pile. Les m composantes du vecteur P sont notées P(1) , P(2),..., P(m) et les composantes P(j) , ptr $\leq$ j $\leq$ m , doivent être considérées comme vides.

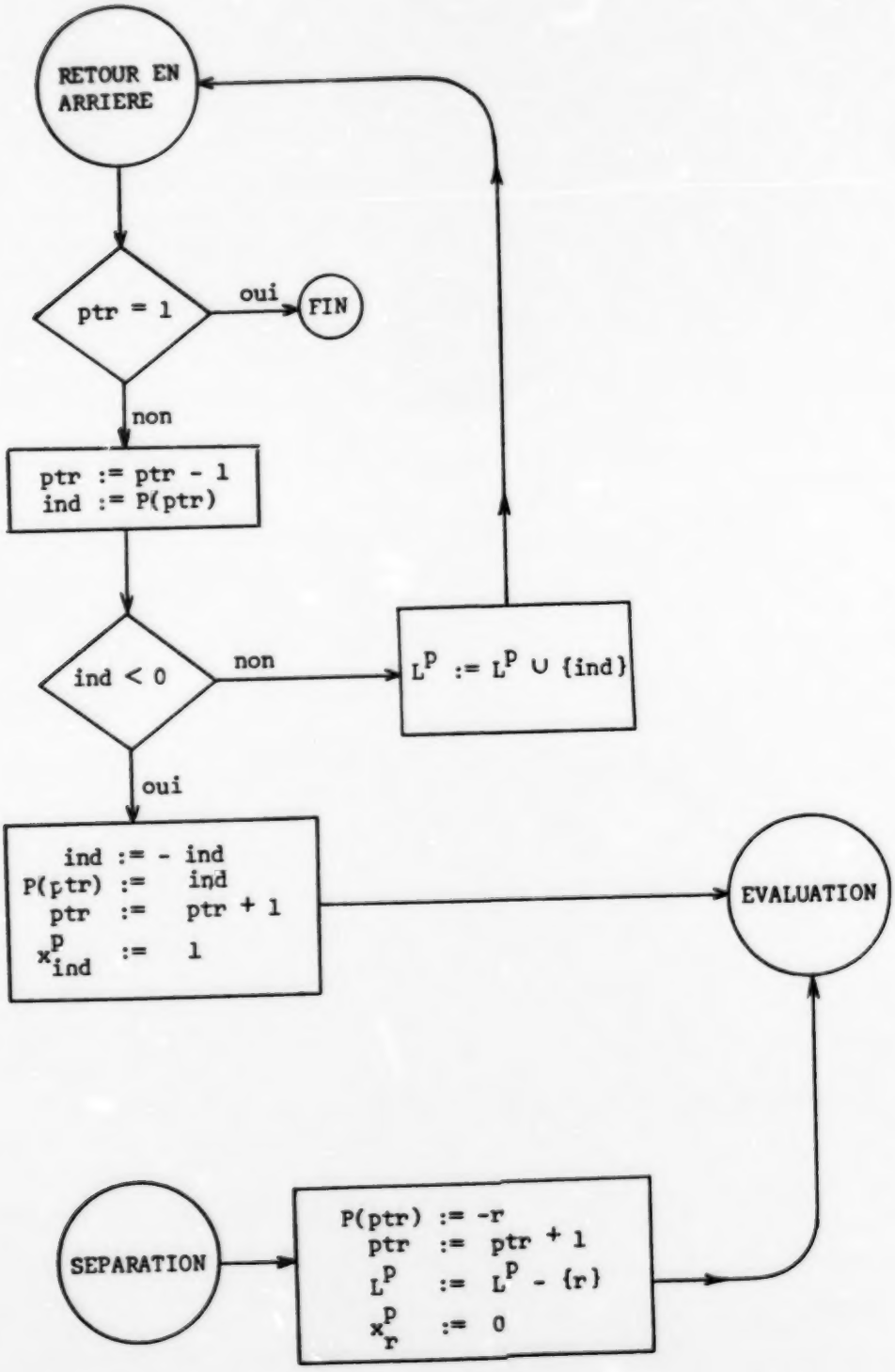
Exemple: dans cette notation, l'exemple de la page 9 s'écrit:

P = (-5, 3, -7, 4, -, -, ..., -)

ptr = 5 .



ORGANIGRAMME DE L'ALGORITHME DE HOANG



ORGANIGRAMME (suite)

2.8 Quelques remarques d'ordre pratique sur l'implantation de l'algorithme

La définition de l'algorithme est peu explicite sur la façon de mener certains calculs. En utilisant quelques astuces (et nous pensons ici à la programmation), on pourra épargner beaucoup de temps de calcul.

Remarque 1: Une façon économique de déterminer si le réseau X^p est connexe consiste à lui associer un vecteur DEG à n composantes, qu'on tiendra à jour tout au long des calculs.

DEG(i) = le degré du sommet i dans le réseau partiel X^p, $i \in N$. Le degré d'un sommet i est défini comme le nombre d'arêtes qui ont i comme extrémité. Initialement, on pose DEG(i) égal au degré du sommet i dans le réseau R, $\forall i \in N$. Ensuite, chaque fois qu'on supprime une arête $a_k = (i,j)$, c'est-à-dire chaque fois que la valeur de x_k^p passe de 1 à 0, on soustrait 1 à DEG(i) et DEG(j). Inversement, DEG(i) et DEG(j) augmentent de 1 si la valeur de x_k^p passe de 0 à 1. Si on procède ainsi, il est clair que la proposition suivante est vraie.

Proposition: $\exists i_o \in N \ni DEG(i_o)=0 \Longrightarrow$ le réseau X^p est disconnexe.

Cependant la contraposée de cette proposition n'est pas vraie en général. Pour identifier que X^p est disconnexe, il suffira souvent de tester si le vecteur DEG comporte une composante nulle. Cependant, si aucune composante de DEG n'est nulle, cela n'implique pas la connexité du réseau X^p et il faudra en vérifier au long la connexité (tous les éléments de la matrice $C(X^p) < \infty$).

Remarque 2: Si $a_k = (i,j)$, évaluer $f_k(X^p)$ exige qu'on calcule la longueur de la plus courte chaîne dans le réseau $(X^p - e_k)$. Pour ce calcul, l'algorithme de Dijkstra (1959) est le plus efficace.

Remarque 3: Dans l'exploration de l'arborescence associée, lorsqu'on effectue un retour en arrière, il arrive qu'on ait à recalculer des $f_j(X^p)$ ou

encore des $c_{ij}(X^p)$ déjà calculés. Pour éviter de refaire ces calculs déjà faits, on serait avisé de conserver en mémoire les valeurs des $f_j(X^p)$ déjà calculées; cela nécessite un nombre de mots de mémoire au plus égal à $m\,n$ (borne théorique), mais en fait beaucoup plus petit. Quant aux $c_{ij}(X^p)$, conserver tous ceux qui sont susceptibles d'être réutilisés exigerait trop d'espace-mémoire; l'astuce du programmeur a ici sa place: il pourra, par exemple, ne conserver que les $c_{ij}(X^p)$ qui changent effectivement d'une évaluation à la suivante.

Remarque 4: Finalement, nous voulons insister sur un aspect important des calculs. Dans l'algorithme tel que nous l'avons présenté, l'évaluation de $F(X^p)$ et des $f_j(X^p)$ dans le but de calculer la borne I^p se fait à chaque sommet de l'arborescence et elle exige un effort de calcul très important.

Pour réduire cet effort, on peut se contenter, comme le suggère Hoang (1971), de faire l'évaluation des $f_j(X^p)$ seulement à chaque fois qu'un certain nombre d'arêtes ont été exclues du réseau X^p, c'est-à-dire suivant une certaine fréquence (notation: FREQ). Au fur et à mesure de l'exploration de l'arborescence associée, on compte le nombre d'arêtes exclues (NEX) et on procède à l'évaluation des $f_j(X^p)$ seulement si

$$NEX \equiv 0 \pmod{FREQ}.$$

Hoang (1971) suggère aussi de ne pas poursuivre l'évaluation des $f_j(X^p)$ au-delà d'un certain niveau (niveau ≡ nombre de variables fixées) dans l'arborescence associée; nous appelons ce niveau NMAX.

L'évaluation des $f_j(X^p)$ exige la connaissance préalable de $F(X^p)$; on se contentera donc d'évaluer $F(X^p)$ lorsque $NEX \equiv 0 \pmod{FREQ}$ et lorsque X^p devient une solution réalisable (pour comparer $F(X^p)$ à $F(X^o)$).

En procédant ainsi, il arrive qu'on ne dispose pas des valeurs de $F(X^p)$ et des $f_j(X^p)$ pour calculer la borne I^p. On utilise alors à leur place les valeurs de F et des f_j les plus récemment calculées (ou récupérées de la mémoire si on procède comme indiqué à la remarque 3, ci-dessus).

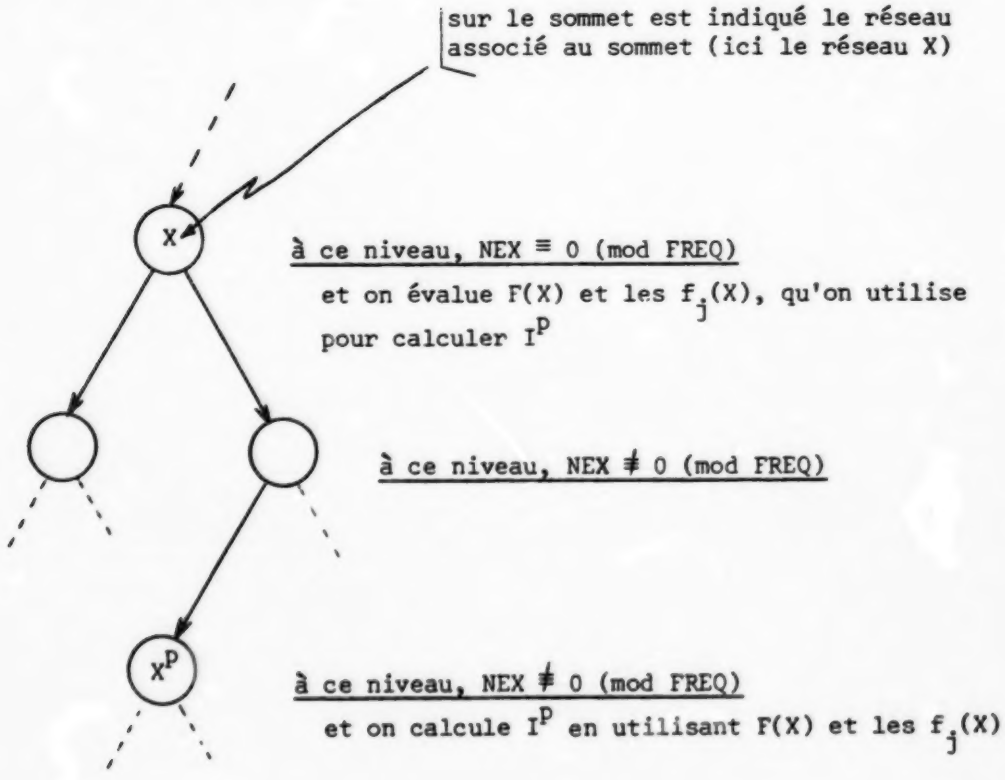


Figure 2 (exemple d'arborescence associée)

Si la plus récente évaluation de F et des f_j a été faite pour le réseau X, par exemple (voir figure 2), on a:

$$F(X) \leq F(X^P) \quad \text{et} \quad f_j(X) \leq f_j(X^P) \quad (j \in M) .$$

Et on calcule I^P en utilisant $F(X)$ à la place de $F(X^P)$ et $f_j(X)$ au lieu de $f_j(X^P)$.

La borne inférieure I^P devient ainsi plus faible et donc moins apte à effectuer une coupe, ce qui entraîne une augmentation du nombre de sommets de l'arborescence associée. Cependant, on espère réaliser une épargne en temps de calcul, étant donné qu'on évalue moins fréquemment F et les f_j. Le choix de FREQ et de NMAX est une question d'heuristique. Nous en reparlerons dans la troisième partie du présent rapport, qui traite de la recherche expérimentale d'un algorithme plus efficace.

3. Une approche expérimentale à la recherche d'un algorithme plus efficace

3.1 Introduction

Le présent travail s'est trouvé amorcé par une idée très simple: dans l'algorithme de Hoang (1971), il paraît plausible que le remplacement de l'algorithme de Floyd-Murchland (défini par Floyd (1962)) par l'algorithme AS (défini et étudié par Dionne (1973)) permette de diminuer le temps de calcul. Dans un réseau symétrique X dont on connaît la matrice C(X) , l'algorithme AS permet de repérer les plus courtes chaînes qui sont effectivement modifiées par la suppression d'une arête et de calculer leur nouvelle longueur plus efficacement que l'algorithme de Floyd-Murchland.

Cependant, dans l'algorithme de Hoang (1971), le calcul de la matrice C n'est pas effectué chaque fois qu'on supprime une arête, tandis que l'algorithme AS , lui, doit être mis en oeuvre après chaque suppression. Pour cette raison, on ne peut déterminer théoriquement avec certitude si le remplacement de l'algorithme de Floyd-Murchland par l'algorithme AS entraîne effectivement une diminution appréciable du temps de calcul. Nous avons donc eu recours à l'expérimentation pour trancher la question.

En cours de travail, nous avons vu d'autres possibilités, nous avons eu d'autres idées susceptibles de donner un algorithme encore plus efficace. Mais toujours la situation se présentait de la même façon: l'implantation d'une idée que nous estimions susceptible de diminuer le temps de calcul entraînait une modification de l'effort de calcul dont il était difficile, sinon impossible, d'évaluer l'impact sur le temps de

calcul; et il fallait recourir à l'expérimentation. Nous avons donc écrit (en Fortran FTN pour l'ordinateur CDC CYBER 74) un programme comportant diverses options de calcul, une option par idée à tester. Comme données, nous disposions de huit réseaux, brièvement décrits dans le tableau 1 (section 3.2).

Les options de calcul que nous avons évaluées et comparées entre elles sont:

- option 1 : l'algorithme de Hoang (solution initiale par la méthode du graphe partiel connexe minimal)
- option 2 : la même que l'option 1, mais on a remplacé l'algorithme de Floyd-Murchland par l'algorithme AS
- option 3 : la même que l'option 2, sauf que le critère de choix de la variable séparatrice devient $(f_j/\ell_j, 1)$
- option 4 : la même que l'option 2, sauf qu'on prend $(Q_j/\ell_j, 1)$ comme critère de choix.

D'une option à la suivante, nous avons cherché à augmenter l'efficacité de l'algorithme initial (option 1), qui semble être lui-même une bonne méthode optimale (d'après Hoang (1971): "[...] the method [...] compares well with the existing optimal methods.").

Nous avons aussi mené une courte étude sur les méthodes d'obtention d'une solution initiale, d'où sont sortis des résultats très intéressants.

3.2 Quelques remarques et notation supplémentaire

Il sera utile pour les fins de la discussion qui suit de donner ici quelques nouvelles définitions.

A un réseau donné, nous associons deux quantités, LMIN et LMAX, qui sont des caractéristiques du réseau, au même titre que m et n :

- LMIN = longueur totale des arêtes faisant partie du graphe partiel connexe minimal
- LMAX = longueur totale des arêtes dans $A = \sum_{j=1}^{m} \ell_j$.

Les huit réseaux que nous avons utilisés ont les caractéristiques suivantes:

nom du réseau	n	m	LMIN	LMAX	provenance
RH1	7	16	122	479	Hoang (1971), pp. 15-16 (longueur des arêtes multipliée par 2)
RH2	8	18	281	1041	
RH3	9	20	298	1122	
RB4	10	45	1654	20619	Boyce *et al.* (1973),p.531
R5	12	20	667	1944	réseaux que nous avons construits
R6	20	40	197	589	
R7	20	40	872	2722	
R8	29	54	1453	3654	

Tableau 1

Pour une option de calcul donnée, nous considérerons dorénavant FREQ et NMAX comme des paramètres de cette option de calcul. De la même façon, nous considérerons B comme un paramètre (paramètre du problème). Cependant, plutôt que de travailler avec le paramètre B (budget), nous trouvons plus pratique d'utiliser un autre paramètre, IB , qui indique quel pourcentage du coût (longueur) maximum LMAX est accordé comme budget:

$$IB = 100 * B / LMAX .$$

En permettant, pour chaque option de calcul, la variation des paramètres, nous aurions généré un trop grand nombre de résultats numériques. Pour éviter cela, nous avons fixé a priori (mais en tenant compte d'une courte expérimentation préalable) la valeur de chacun des trois paramètres impliqués dans notre étude, en posant:

- FREQ = 3 , parce que l'expérimentation préalable avait indiqué que l'algorithme de Hoang est plus rapide pour de petites valeurs de FREQ
- NMAX > m , ce qui rend inopérant le paramètre NMAX , sur l'utilité duquel l'expérimentation préalable nous avait laissés fort sceptiques
- IB = 50 .

En ne donnant ainsi qu'une seule et même valeur à chaque paramètre, quels que soient l'option de calcul et le réseau considérés, nous facilitions grandement le travail de comparaison des résultats numériques obtenus avec les différentes options de calcul. Nous étions bien conscients cependant que cela comportait un danger: le choix de ces valeurs pour les para-

mètres pouvait favoriser une option de calcul aux dépens des autres. D'où la nécessité de faire (a posteriori) une analyse paramétrique des résultats, dont on trouvera un compte-rendu à la section 3.5.

Pour faciliter l'analyse des résultats numériques dans les pages qui suivent, nous les désignons par des identificateurs, dont la définition apparaît dans la table ci-dessous. On ne devrait pas avoir à consulter trop souvent cette table, car les identificateurs ont été choisis pour leur qualité mnémotechnique.

3.2.1 Table des identificateurs

- NSOM : désigne le nombre de sommets dans l'arborescence associée générée par l'option de calcul
- NAS : nombre de fois où l'algorithme AS est mis en oeuvre
- NFLOYD : nombre de fois où l'algorithme de Floyd-Murchland est mis en oeuvre
- NEVAL : nombre de fois où l'on procède à l'évaluation des f_j
- NBORNE : nombre de fois où l'on calcule la borne I^P
- UTIL% : mesure (en pourcentage) de l'utilité de la borne I^P, c'est-à-dire de sa capacité à effectuer une coupe:
  UTIL% = 100 * (nombre de fois où la borne coupe) / NBORNE
- MCOUPE : niveau moyen auquel la borne I^P effectue une coupe:
  $$\text{MCOUPE} = \frac{\text{somme des niveaux auxquels la borne coupe}}{\text{nombre de fois où la borne coupe}}$$
- NOPT : numéro du sommet où la solution optimale a été trouvée dans l'arborescence associée
- T : temps de calcul en secondes (à l'exclusion du temps pris pour calculer la solution initiale)
- T_{in} : temps de calcul de la solution initiale
- Z : valeur de la fonction économique à l'optimum
- Z_{in} : valeur de la fonction économique pour la solution initiale

Comme la donnée d'un réseau et d'une valeur pour le paramètre IB suffit à définir un problème de choix optimal d'un réseau, nous désignerons ci-après un tel problème par la formule (réseau, valeur de IB). Par exemple, (RH1,50).

3.3 Comparaison de l'option 1 et de l'option 2

Nous avons déjà expliqué, au tout début de la section 3, pourquoi nous espérions réduire le temps de calcul en remplaçant, dans l'algorithme de Hoang, l'algorithme de Floyd-Murchland (option 1) par l'algorithme AS (option 2). Nous donnons immédiatement les résultats numériques obtenus pour chacune de ces deux options.

problème → / ↓identificateur		(RH1,50)	(RH2,50)	(RH3,50)	(RB4,50)	(R5,50)
NSOM	option 1	451	1853	4431	177	733
	option 2	209	938	1928	155	144
NFLOYD →	option 1	113	426	993	23	225
NAS →	option 2	80	365	775	52	62
NEVAL	option 1	19	111	146	18	2
	option 2	17	85	123	16	2
NBORNE	option 1	270	1131	2665	117	411
	option 2	129	573	1150	103	82
UTIL%	option 1	33.0	36.2	33.7	48.7	21.7
	option 2	38.0	36.3	32.3	49.5	24.4
MCOUPE	option 1	9.0	10.9	12.6	8.1	12.3
	option 2	8.2	9.7	10.6	7.9	8.8
NOPT	option 1	120	1418	831	15	6
	option 2	63	725	425	15	6
T(s.)	option 1	0.90	513	12.12	5.03	2.47
	option 2	0.47	2.94	5.74	4.38	0.53

Rappel: FREQ = 3 et NMAX > m .

Remarque: Les réseaux R6, R7 et R8 ne sont pas considérés ici à cause de leur temps de calcul prohibitif. Il en sera de même dans les tableaux 3,4 et 5.

Tableau 2

Dans le tableau 2, on aura remarqué que NSOM(2)* < NSOM(1) . Mais pourquoi l'option 2 génère-t-elle ainsi une arborescence associée qui contient moins de sommets que l'option 1? Il n'y a qu'une explication possible. A chaque sommet de l'arborescence associée, correspond une solution X^P . Or l'option 1 évalue la matrice $C(X^P)$ et, à partir d'elle, $F(X^P)$ dans deux cas seulement: avant d'évaluer les f_j et lorsque X^P est une solution réalisable. Ainsi, la valeur exacte de $F(X^P)$ peut ne pas être disponible, dans l'option 1; on utilise alors à sa place la valeur la plus récemment calculée de la fonction économique F , disons F_a ($F_a \leq F(X^P)$) . Comme $F(X^P)$ constitue l'un des termes de la borne inférieure I^P , il en résulte pour l'option 1 une sous-évaluation de la borne, qui peut ne pas entraîner une coupe, alors que la valeur exacte de la borne aurait effectué une coupe. Si on compare UTIL%(1) et UTIL%(2) dans le tableau 2, on constate que la borne I^P coupe à peu près aussi fréquemment dans l'option 1 que dans l'option 2. Cependant, les coupes s'effectuent en moyenne plus bas dans l'arborescence associée pour l'option 1 que pour l'option 2 (MCOUPE(1) > MCOUPE(2)). Il en résulte que NSOM(1) > NSOM(2) , NBORNE(1) > NBORNE(2) et NEVAL(1) > NEVAL(2) pour tous les problèmes que nous avons traités.

On ne peut pas corriger ce défaut inhérent à l'option 1 (sous-évaluation de I^P) en calculant $C(X^P)$ à chaque sommet exploré, parce que l'algorithme de Floyd-Murchland est trop lent. Dans l'option 2, ce défaut est évité, car la matrice C est constamment tenue à jour par l'algorithme AS .

On aura aussi remarqué que l'utilité de la borne est de l'ordre de

* Note: La notation "NSOM(2)" signifie "NSOM pour l'option 2".

33%. C'est donc dire que calculer la borne I^P conduit une fois sur trois environ à effectuer une coupe, ce qui est considérable.

Expérimentalement, l'option 2 se révèle nettement plus efficace que l'option 1. En effet (à une exception près (NFLOYD(1) < NAS(2) pour RB4), qui est sans importance, puisque l'algorithme de Floyd-Murchland est plus lent que AS), tous les résultats numériques de l'option 2 sont plus intéressants que ceux de l'option 1, dans le tableau 2. Nous ne conservons donc que l'option 2, que nous allons maintenant comparer à l'option 3.

3.4 Comparaison de l'option 2 et de l'option 3

L'option 3 est obtenue de l'option 2 par simple modification du critère de choix de la variable séparatrice: dans l'option 3, on prend x_1 comme variable séparatrice plutôt que x_r comme dans l'option 2.

Cette modification se justifie facilement. On se rappellera qu'avant de résoudre le problème (2) (voir section 2.4), on réordonne les variables non encore fixées de façon à obtenir

$$\frac{f_1(X^P)}{\ell_1} < \frac{f_2(X^P)}{\ell_2} < \ldots < \frac{f_e(X^P)}{\ell_e} \quad ,$$

$$\text{alors que} \quad L^P = \{1,2,\ldots,e\} \quad .$$

Hoang suggère de choisir x_r comme variable séparatrice (r , le plus petit entier $\ni$ $0 < r \leq e$ et $\sum_{i=1}^{r} \ell_i > b$) .

Nous voyons mal la motivation de son choix. En effet, si l'arête a_k s'écrit $a_k = (i,j)$, f_k représente l'allongement subi par la plus courte chaîne de i à j lorsqu'on supprime a_k . On peut donc considérer, que f_k/ℓ_k représente une mesure (relative et assez peu précise) de l'allongement par unité de longueur causé par la suppression de l'arête a_k . Dans ce cas, il serait plus approprié de choisir x_1 comme variable séparatrice, puisqu'à x_1 correspond la plus petite mesure d'allongement unitaire (f_1/ℓ_1) . Cette mesure, bien qu'imprécise, constitue un indice pouvant nous guider plus rapidement vers une solution optimale ou en tout cas vers des solutions réalisables plus proches de l'optimum, d'où une meilleure évaluation de la borne I^p et, par suite, une arborescence générée plus petite et un temps de calcul plus court. Voyons si les résultats numériques confirment nos dires.

problème → / ↓identificateur		(RH1,50)	(RH2,50)	(RH3,50)	(RB4,50)	(R5,50)
NSOM	option 2	209	938	1928	155	144
	option 3	134	781	188	62	234
NAS	option 2	80	365	775	52	62
	option 3	56	297	72	23	89
NEVAL	option 2	17	85	123	16	2
	option 3	8	69	17	6	19
NBORNE	option 2	129	573	1150	103	82
	option 3	78	483	115	39	145
UTIL%	option 2	38.0	36.3	32.3	49.5	24.4
	option 3	28.2	38.3	36.5	41.0	38.6
MCOUPE	option 2	8.2	9.7	10.6	7.9	8.8
	option 3	7.6	8.8	7.8	8.9	6.8
NOPT	option 2	63	725	425	15	6
	option 3	88	12	46	32	6
T(s.)	option 2	0.47	2.94	5.74	4.38	0.53
	option 3	0.30	2.19	0.70	1.60	1.08

Rappel: FREQ = 3 et NMAX > m .

Remarque: Nous reprenons ici les résultats du tableau 2 pour l'option 2 afin de faciliter le travail de comparaison.

Tableau 3

Pour les réseaux RH1, RH2, RH3 et RB4, les résultats numériques correspondent assez bien à ce que nous attendions de l'option 3:

- un optimum atteint plus rapidement (NOPT(3) << NOPT(2)), lorsque NOPT(2) est grand; et un optimum atteint après l'examen d'un nombre semblable de sommets (NOPT(3) ≅ NOPT(2)) , lorsque NOPT(2) est petit
- une arborescence associée plus petite (NSOM(3) < NSOM(2))
- et, par conséquent, un temps de calcul plus court (T(3) < T(2)) .

Pour le problème (R5,50) , l'option 3 ne fait pas mieux que l'option 2, mais il faut remarquer que l'arborescence associée générée par chacune des deux options est relativement petite (NSOM(2) = 144 , NSOM(3) = 234) et que NOPT(2) = NOPT(3) = 6 .

En suggérant de remplacer le critère de choix de la variable séparatrice $(f_j/\ell_j , r)$ par le critère $(f_j/\ell_j , 1)$, nous proposons une nouvelle heuristique, sans nous attendre à ce qu'elle soit meilleure dans tous les cas. Nous croyons cependant que la nouvelle heuristique proposée est, dans l'ensemble, meilleure que l'option 2. Et les résultats du tableau 3 nous confirment dans cette opinion. Il peut sembler que nous concluons à partir d'un petit nombre de résultats numériques, mais les résultats du tableau 3 ne sont pas les seuls que nous ayons recueillis: nous avons aussi obtenu des résultats pour les 20 problèmes définis par les réseaux RH1, RH2, RH3, RB4 et R5 et les budgets IB = 30, 40, 60 et 70. Ce sont toujours les mêmes tendances qui se dégagent. C'est pourquoi nous conservons l'option 3 comme la meilleure option jusqu'à maintenant.

3.5 Etude paramétrique de l'option 3

Pour atteindre une meilleure compréhension de l'option 3, il reste à étudier les paramètres FREQ et NMAX . Observons d'abord les effets du paramètre FREQ sur le comportement de l'algorithme.

	problème →	(RH1,50)	(RH2,50)	(RH3,50)	(R5,50)
NMAX =	FREQ = 1	(NSOM=134,NEVAL=22) T =0.45	(693,170) 3.64	(176,42) 1.29	(228,54) 1.98
	FREQ = 2	(134,11) 0.32	(747,92) 2.52	(184,24) 0.87	(239,29) 1.37
	FREQ = 3	(134,8) 0.30	(781,69) 2.19	(188,17) 0.72	(234,19) 1.08
	FREQ = 4	(137,5) 0.28	(806,39) 1.93	(189,9) 0.57	(262,20) 1.10
	FREQ = 5	(134,7) 0.27	(824,62) 2.17	(188,10) 0.57	(274,9) 0.90
	FREQ = 6	(134,5) 0.24	(851,49) 1.95	(188,12) 0.56	(268,1) 0.74
	FREQ = 7	(137,1) 0.22	(885,28) 1.72	(188,3) 0.45	
	FREQ = 8		(907,1) 1.52	(192,2) 0.45	
	FREQ = 9			(192,1) 0.41	

Remarques: 1) Chaque élément du tableau doit se lire: (NSOM, NEVAL) T(s.)

2) Le réseau RB4 n'apparaît pas ici, parce que les résultats sont presque toujours les mêmes quelle que soit la valeur de FREQ .

Tableau 4

L'effet du paramètre FREQ , lorsque NMAX > m , ressort de façon très claire:

lorsque FREQ grandit, NSOM accuse une légère tendance à la hausse, cependant que NEVAL décroît franchement. Même si NSOM grandit un peu, les f_j sont évalués moins souvent et l'effet net apparaît: le temps de calcul T diminue quand FREQ grandit. Pour α suffisamment grand,T reste le même et NEVAL devient égal à 1 pour toute valeur de FREQ $\geq \alpha$.

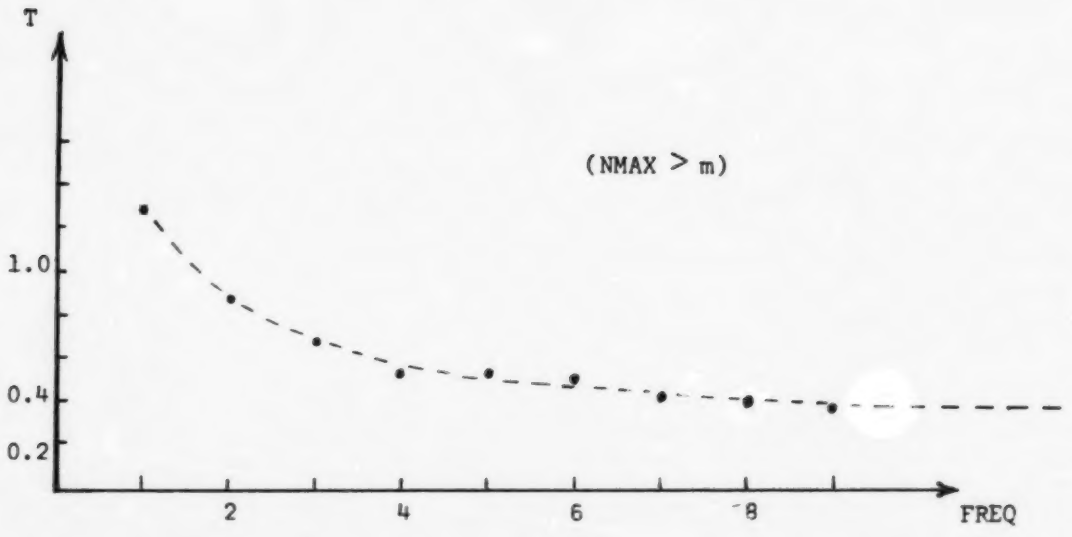


Figure 3

Graphique (T vs FREQ) pour l'option 3

(Note: Les valeurs numériques portées sur le graphique sont celles du tableau 5 pour le problème (RH3,50), mais l'allure de la courbe est la même pour n'importe quel problème.)

Les paramètres FREQ et NMAX jouent un rôle utile dans l'algorithme de Hoang (option 1). Mais l'option 3 diffère considérablement de l'option 1: les paramètres FREQ et NMAX n'y ont aucun rôle à jouer, puisque T atteint son minimum lorsqu'on évalue les f_j une seule fois avant de commencer à générer l'arborescence associée (NEVAL = 1) . Dès lors, nous supprimons les deux paramètres FREQ et NMAX de l'option 3.

Il y a un troisième paramètre, dont il est intéressant de connaître l'effet sur le temps de calcul: c'est le paramètre IB (paramètre du problème).

réseau →	RH1	RH2	RH3	RB4	R5
IB = 70	0.11	0.27	0.29	1.81	0.45
IB = 60	0.08	0.45	0.46	3.10	0.74
IB = 50	0.30	2.19	0.70	1.60	1.08
IB = 40	0.33	1.67	1.37	1.79	3.08
IB = 30	1.43	5.38	14.84	10.03	(LMIN > B)

T (s.) en fonction de IB

Tableau 5

Un fait ressort clairement: le temps de calcul augmente lorsque IB diminue. Boyce *et al.* (1973) ont d'ailleurs obtenu un résultat semblable pour leur algorithme.

Le type d'algorithme représenté par les diverses options de calcul que nous étudions se comporte bien lorsque le budget est peu restrictif, mais il lui faut beaucoup de temps pour résoudre un problème à petit budget. En effet, pour un petit budget, l'algorithme doit éliminer plusieurs arêtes du réseau initial R avant d'atteindre une solution X^P qui satisfasse la contrainte du budget; il doit donc descendre à un niveau plus

bas (i.e. un niveau de numéro plus grand) en générant l'arborescence associée. Or, dans une arborescence binaire à n sommets, si on ajoute 2 sommets comme successeurs à chaque sommet pendant, on se trouve à multiplier par 2 le nombre de sommets dans l'arborescence (plus précisément, cela donne $2n + 1$ sommets). D'une façon analogue (pour l'arborescence associée), si on diminue le budget suffisamment, on descent 1 niveau plus bas en générant l'arborescence associée et on multiplie par 2 le nombre de sommets générés. Par suite, le temps de calcul (grosso modo proportionnel au nombre de sommets générés) a tendance à être proportionnel à

$$2^{-c * IB} \qquad \text{(c, une constante positive).}$$

Bien sûr, il ne s'agit là que d'une tendance, puisque la contrainte de connexité limite la génération des sommets et qu'il y a d'autres facteurs qui ont une influence sur le temps de calcul (par exemple, les sommets qu'on atteint après avoir posé une variable égale à 0 exigent plus de temps de calcul que ceux qu'on atteint après avoir fixé une variable égale à 1). Ce comportement exponentiel du temps de calcul est difficile à mettre en évidence numériquement pour de petits problèmes comme ceux du tableau 5, mais nous y reviendrons à la section 3.8 et nous le mettrons en évidence.

3.6 Comparaison de l'option 3 et de l'option 4

Nous avons vu à la section 2.1 que, pour un réseau donné X, $Q_k(X)$ constitue une meilleure mesure de l'utilité de l'arête a_k que $f_k(X)$. Or, dans l'option 3, les f_j jouent deux rôles importants:

- par l'intermédiaire du critère de choix $(f_j/\ell_j, 1)$, ils servent à déterminer le choix de la variable séparatrice et ils constituent ainsi un guide permettant d'atteindre plus rapidement la solution optimale
- et ils entrent dans le calcul de la borne I^P, qui contribue fortement à réduire le nombre de sommets dans l'arborescence associée.

Il serait donc souhaitable de faire jouer ces deux rôles par les Q_j, d'autant plus que l'algorithme AS peut être mis à contribution pour calculer rapidement les Q_j.

Les Q_j peuvent facilement remplacer les f_j dans la définition du critère de choix de la variable séparatrice: il suffit de prendre $(Q_j/\ell_j, 1)$ comme critère de choix. (Précisons bien ici que cette modification du critère de choix ne doit pas affecter la façon de calculer la borne I^P : pour le calcul de la borne, les variables continuent d'être ordonnées suivant l'ordre croissant des f_j/ℓ_j.)

Quant à l'utilisation des Q_j à la place des f_j dans le calcul de la borne I^P, elle s'avère impraticable. En effet, le théorème I n'est plus vrai si on y remplace f_j par Q_j.

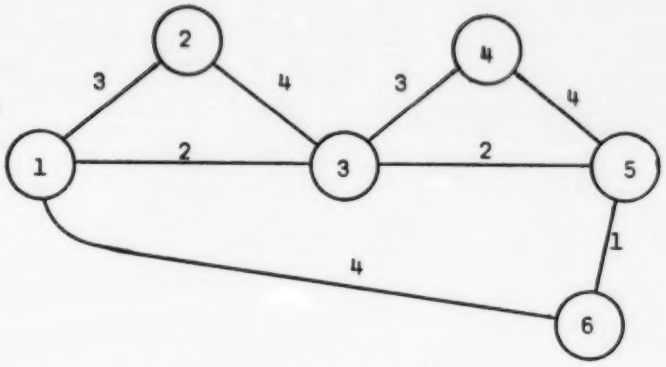
<u>Enoncé du théorème I (rappel)</u>: $\forall X \in E^P$, on a:

$$F(X) \geq F(X^P) + \sum_{j \in L^P} \bar{x}_j \, f_j(X^P) .$$

<u>Enoncé à infirmer</u>: $\forall X \in E^P$, on a:

$$F(X) \geq F(X^P) + \sum_{j \in L^P} \bar{x}_j \, Q_j(X^P) .$$

Contrexemple:



Numérotons les arêtes:

$a_1 = (1,2)$	$\ell_1 = 3$
$a_2 = (1,3)$	$\ell_2 = 2$
$a_3 = (2,3)$	$\ell_3 = 4$
$a_4 = (3,4)$	$\ell_4 = 3$
$a_5 = (3,5)$	$\ell_5 = 2$
$a_6 = (4,5)$	$\ell_6 = 4$
$a_7 = (1,6)$	$\ell_7 = 4$
$a_8 = (5,6)$	$\ell_8 = 1$

Soient $X^p = (1,1,1,1,1,1,1,1)$

$X' = (1,0,1,1,1,1,1,1)$

$X'' = (1,0,1,1,0,1,1,1)$

$F(X^p) = 40$ et $Q_2(X^p) = Q_5(X^p) = 11$

$F(X'') = 61 < F(X^p) + Q_2(X^p) + Q_5(X^p) = 62$

Fin du contrexemple.

Si l'on calcule la borne I^p en utilisant les Q_j à la place des f_j, il peut arriver que la borne I^p surestime la valeur de la fonction

économique de certaines solutions de E^p et on risque ainsi d'effectuer une coupe qui fera perdre la solution optimale. Nous utilisons cependant la borne I^p calculée à l'aide des Q_j dans une heuristique très efficace, qui calcule une solution réalisable toujours très proche de l'optimum et même presque toujours égale à l'optimum, et dont nous reparlerons plus loin (cf. sections 3.7 et 3.8, méthode MQO) . Cette nouvelle façon de calculer fournit une borne inférieure I^p beaucoup plus forte et donc beaucoup plus apte à couper l'arborescence associée, qui comporte alors beaucoup moins de sommets; d'où un temps de calcul réduit considérablement.

Nous devons donc nous contenter, dans notre tentative d'améliorer l'option 3, de faire jouer par les Q_j un seul des deux rôles joués par les f_j dans l'option 3. Nous proposons donc l'option 4, qui diffère de l'option 3 par le critère de choix de la variable séparatrice, qui devient $(Q_j/\ell_j , 1)$.

Puisque les Q_j constituent une meilleure mesure de l'utilité des arêtes que les f_j , nous nous attendons intuitivement à ce que les Q_j/ℓ_j se révèlent meilleurs guides que les f_j/ℓ_j dans le choix de la variable séparatrice et qu'ils nous conduisent plus rapidement vers l'optimum, entraînant ainsi une réduction du temps de calcul. Mais l'expérimentation a montré qu'il en va autrement, ainsi qu'on peut le constater en consultant le tableau 6. S'il est vrai que l'optimum a tendance à être atteint plus rapidement par l'option 4 (NOPT(4) du même ordre de grandeur ou bien plus petit que NOPT(3)), cela n'entraîne pas une diminution du temps de calcul pour l'option 4, étant donné que par la même

occasion, l'arborescence associée grandit (NSOM(4) > NSOM(3)) . Les arêtes ne sont pas éliminées dans le même ordre par le critère de choix de l'option 4 et celui de l'option 3. Or la valeur du terme $F(X^P)$ et la valeur du terme MIN $\sum_{j\in L^P} \bar{x}_j f_j(X^P)$ de la borne I^P sont toutes deux affectées par le choix des arêtes éliminées. Tout se passe comme si une plus grande concordance entre la mesure qui sert à définir le critère de choix de la variable séparatrice (i.e. f_j/ℓ_j pour l'option 3 et Q_j/ℓ_j pour l'option 4) et la mesure qui sert à calculer la borne I^P (i.e. f_j) contribuait à rendre la borne I^P plus apte à couper l'arborescence associée; cette concordance est plus grande pour l'option 3. L'explication que nous venons de donner est sûrement partielle, mais le phénomène échappe à l'analyse à cause de la complexité des interactions en jeu au cours des calculs.

L'option 3 demeure la meilleure option de calcul. C'est la meilleure méthode optimale que nous ayons mis au point.

	option 3				option 4			
↓problème	NSOM	MCOUPE	NOPT	T (s.)	NSOM	MCOUPE	NOPT	T (s.)
(R6,65)	2498	14.9	1758	19.2	7856	15.9	1792	75.4
(R6,60)	3623	15.8	2647	27.9	10700	15.9	9	99.5
(R6,55)	4646	15.1	2619	35.2	22527	16.9	113	208.3
(R7,65)	530	9.9	9	3.7	1961	11.5	9	17.0
(R7,62)	3622	15.1	27	25.9	14856	16.1	19	135.2
(R7,60)	8363	17.4	107	61.5	43637	18.2	178	405.8
(R8,80)	1675	13.9	368	24.8	18902	16.8	10	338.1

Remarques: 1) Nous donnons ici des résultats obtenus pour les réseaux R6, R7 et R8. Pour les 5 autres réseaux (plus petits), les résultats ne diffèrent pas de façon significative de l'option 3 à l'option 4.

2) Nous rappelons que les paramètres FREQ et NMAX ont été éliminés de l'option 3. Après avoir vérifié que leur réintroduction n'améliore aucunement l'option 4, nous ne les avons pas non plus employés dans l'option 4.

Tableau 6

3.7 Evaluation des méthodes d'obtention d'une solution initiale

Nous avons décrit à la section 2.6 deux méthodes pour déterminer une solution initiale:

- la méthode du graphe partiel connexe minimal (ci-après, méthode MGPM)
- et la méthode des Q_j (ci-après, méthode MQJ).

Nous proposons une troisième méthode, que nous appellerons méthode MQO (i.e. méthode quasi optimale) et qui n'est autre que l'option 4 modifiée de façon à ce que la borne I^p soit calculée en utilisant les Q_j à la place des f_j .

Nous nous proposons maintenant d'évaluer chacune de ces trois méthodes. Nos critères d'évaluation sont le temps de calcul (T_{in}) de la solution initiale fournie par la méthode, ainsi que l'écart relatif entre la solution initiale et la solution optimale mesuré par

$$Er = \frac{Z_{in} - Z}{Z_{in}} .$$

Pour la méthode MGPM , nous obtenons les résultats suivants:

- le temps de calcul est négligeable (et toujours le même quelque soit le budget)
- mais la solution initiale fournie est somme toute très mauvaise.

Les deux autres méthodes fournissent une solution initiale beaucoup plus proche de l'optimum. La méthode MQO se montre supérieure à la fois pour le temps de calcul T_{in} et pour l'écart relatif Er , du moins dans le cas des problèmes considérés ici. La méthode MQO fournit en un temps très court une solution initiale qui coïncide presque toujours avec la solution optimale (une seule exception pour 18 problèmes traités: pour le problème (R7,60) , l'écart Er = 0.03%) .

		méthode MGPM			méthode MQJ			méthode MQO		
problème	Z	Z_{in}	T_{in}	Er	Z_{in}	T_{in}	Er	Z_{in}	T_{in}	Er
(RH1,50)	761	764	0.03	0.4%	761	0.10	0	761	0.20	0
(RH2,50)	2246	2354	0.03	4.6%	2246	0.19	0	2246	0.63	0
(RH3,50)	2707	2775	0.03	2.5%	2732	0.28	0.9%	2707	0.32	0
(RB4,50)	20696	20873	0.03	0.8%	20700	0.96	0.02%	20696	0.43	0
(R5,50)	11945	12335	0.03	3.2%	11945	0.52	0	11945	0.20	0
(R6,65)	5460	5510	0.08	0.9%	5474	3.86	0.2%	5460	1.14	0
(R6,60)	5478	5659	↓	3.2%	5491	4.26	0.2%	5478	0.69	0
(R6,55)	5503	5707		3.5%	5503	5.46	0	5503	0.69	0
(R6,53)	5521	5707		3.3%	5535	5.45	0.3%	5521	0.80	0
(R6,50)	5563	5806		4.2%	5585	5.75	0.4%	5563	1.83	0
(R7,65)	30705	32254	0.07	4.8%	30871	4.79	0.6%	30705	0.61	0
(R7,62)	30832	32295	↓	4.5%	30832	5.20	0	30832	1.17	0
(R7,60)	30924	32295		4.2%	30932	5.12	0.04%	30932	1.53	0.03%
(R7,58)	31070	32320		3.9%	31070	5.15	0	31070	1.86	0
(R8,80)	99611	103334	0.14	3.6%	99611	12.95	0	99611	1.79	0
(R8,77)	99698	103421	↓	3.6%	99738	12.98	0.4%	99698	3.07	0
(R8,75)	99752	106301		6.2%	99752	14.21	0	99752	1.71	0
(R8,73)	99914	106301		6.0%	99914	15.47	0	99914	2.71	0

Tableau 7

Il est intéressant de noter qu'on peut apporter une modification très simple à la méthode MQJ et l'améliorer ainsi d'une façon appréciable. La modification consiste à utiliser les Q_j/ℓ_j plutôt que simplement les Q_j. L'étape (ii)b) de la méthode MQJ (cf. section 2.6) devient donc:

$$\text{(ii)b) ... trouver } k \ni \frac{Q_k(X)}{\ell_k} = \min_{\{j \in M | x_j = 1\}} \frac{Q_j(X)}{\ell_j} \text{ et poser } x_k = 0 .$$

Cette modification permet de tenir compte du coût (ℓ_j) d'une arête (a_j) en même temps que de sa tendance à faire augmenter la longueur du réseau si on l'enlève (Q_j) ; les Q_j/ℓ_j mesurent l'utilité par unité de longueur et permettent une meilleure comparaison entre les arêtes susceptibles d'être éliminées.

Nous avons utilisé la méthode MQJ modifiée (ci-après, méthode MQJM) pour traiter les 18 problèmes du tableau 7 et nous avons obtenu les résultats suivants:

- légère réduction du temps de calcul T_{in} par rapport à la méthode MQJ (ordre de grandeur de la réduction, en moyenne: 10%)
- la solution initiale obtenue était optimale dans tous les cas sauf 2 (pour (R6,55), $E_r = 0.04\%$ et pour (R8,80), $E_r = 0.003\%$).

Grosso modo, la méthode MQJM semble donc ici aussi apte que la méthode MQO à trouver l'optimum (la plupart du temps) ou, à défaut, une solution réalisable proche de l'optimum. Elle est cependant plus lente que

la méthode MQO pour solutionner les 18 problèmes du tableau 7.

Finalement, nous avons tenté d'améliorer les performances de l'option 3 en lui faisant calculer la solution initiale par la méthode MQO et non plus par la méthode MGPM. Cette modification entraîne en général une diminution du temps de calcul T , diminution qui varie beaucoup d'un problème à l'autre. En effet, la diminution de T est d'autant plus importante que la diminution de NOPT est plus grande. De plus, en faisant subir cette modification à l'option 3, le temps de calcul de la solution initiale (T_{in}) croît légèrement et il peut même arriver qu'aucun gain ne soit obtenu sur le temps de calcul total ($T + T_{in}$) .

Même si l'option 3 s'avère peu sensible à un changement de la solution initiale, nous croyons qu'il vaut mieux en général remplacer MGPM par MQO dans l'option 3, ainsi qu'on pourra le vérifier dans le tableau suivant.

Option 3: solution initiale par MGPM — solution initiale par MQO

problème	NSOM	NOPT	$T_{in}^{T} \cong 0.1$	NSOM	NOPT	$T_{in} + T = \ldots$
(R6,65)	2498	1758	19.2	1437	1	1.2 + 11.1 = 12.3
(R6,60)	3623	2647	27.9	1453	1	0.8 + 10.8 = 11.6
(R6,55)	4646	2619	35.2	2622	1	0.8 + 19.3 = 20.1
(R6,53)	8566	3838	63.9	6229	1	0.9 + 46.7 = 47.6
(R6,50)	37117	9875	279.3	34877	1	1.9 + 266.4 = 268.3
(R7,65)	530	9	3.7	530	1	0.7 + 3.7 = 4.4
(R7,62)	3622	27	25.9	3622	1	1.2 + 26.4 = 27.6
(R7,60)	8363	107	61.5	8284	28	1.6 + 62.2 = 63.8
(R7,58)	26463	546	198.6	25971	1	1.9 + 198.3 = 200.2
(R8,80)	1675	368	24.8	1319	1	1.9 + 19.3 = 21.2
(R8,77)	2973	774	44.6	2261	1	3.2 + 33.5 = 36.7
(R8,75)	3458	1140	52.2	2357	1	1.9 + 34.9 = 36.8
(R8,73)	14745	1616	225.6	13171	1	2.9 + 203.1 = 206.0

Tableau 8

3.8 Les méthodes MQO et MQJM vues comme des heuristiques

Les méthodes MQO et MQJM ne sont pas des algorithmes optimaux, ce sont des heuristiques qui fournissent une très bonne solution réalisable, mais dont l'optimalité n'est pas garantie par la méthode de calcul.

La nature hautement combinatoire du problème du choix optimal d'un réseau constitue un obstacle à l'obtention d'un algorithme optimal qui soit efficace dans la résolution de n'importe quel problème: il y a encore beaucoup de problèmes dont la résolution exige trop de temps de calcul. A la lumière de toute l'expérimentation que nous avons menée, nous croyons pouvoir dire que les facteurs qui contribuent à allonger le temps de calcul se répartissent comme suit:

- 1) Le paramètre IB influence grandement le temps de calcul T . En effet, pour le type d'algorithme optimal que nous avons étudié, T grandit exponentiellement à mesure que IB décroît, ainsi que nous l'avons expliqué à la fin de la section 3.5. Nous reviendrons là-dessus un peu plus loin dans la présente section. Pour l'instant, qu'il nous suffise de renvoyer le lecteur au tableau 8 pour constater comment T croît. Pour un réseau de la taille de R6 ou R7 (20 sommets, 40 arêtes), il suffit de rendre $IB < 50$ pour faire en quelque sorte exploser le temps de calcul exigé par notre meilleur option de calcul (l'option 3).
- 2) Le nombre n est aussi un facteur qui a une influence directe, quoique moins importante que celle de IB , sur le temps de calcul. En effet, à chaque sommet de l'arborescence associée, on calcule la matrice C_{nxn} . Heureusement, dans l'option 3, ce calcul

n'exige plus $O(n^3)$ opérations élémentaires (algorithme de Floyd-Murchland), mais un nombre beaucoup moindre (algorithme AS), que nous estimons, d'après les résultats d'une étude précédente (Dionne (1973)), être voisin de $O(n)$. Même si n était de l'ordre de quelques centaines, le temps de calcul resterait abordable à condition que le nombre de sommets dans l'arborescence associée (NSOM) reste petit, ce qui se produit lorsque le budget n'est pas trop restrictif (i.e. lorsque IB est grand).

- 3) Le nombre d'arêtes, m , a aussi une certaine importance, ainsi que l'illustrera l'exemple suivant. Soient deux réseaux, R et R' , qui comportent le même nombre de sommets, mais supposons que R comporte 2 ou 3 fois plus d'arêtes que R' . Supposons de plus que toutes les arêtes (des 2 réseaux) ont la même longueur. Enfin supposons donnée une valeur pour le paramètre IB (la même pour les 2 réseaux). Alors il est clair qu'en cherchant à résoudre le problème (R,IB) , l'algorithme devra, avant d'obtenir une solution qui satisfasse la contrainte du budget, éliminer plus d'arêtes qu'en cherchant à résoudre (R',IB). D'où une arborescence associée plus grande et un temps de calcul plus long.
- 4) Enfin, la topologie du réseau initial R fait que les plus courtes chaînes sont plus ou moins difficiles à calculer.

Puisque le temps de calcul peut, dans certains cas, devenir prohibitif, des heuristiques comme MQO et MQJM constituent une alternative intéressante pour traiter les problèmes dont le degré de difficulté dépasse les capacités des méthodes optimales. Nous nous proposons mainte-

nant de faire ressortir les avantages respectifs de la méthode MQO et de la méthode MQJM .

La méthode MQO s'est révélée la plus rapide pour traiter les 18 problèmes du tableau 7. Cependant, cette méthode est une P.S.E. et elle devrait être affligée du même défaut que toutes les P.S.E. que nous avons étudiées (options 1 à 4), c'est-à-dire que son temps de calcul devrait être approximativement proportionnel à $2^{-c * IB}$ (cf. fin de la section 3.5). Pour mettre cela en évidence, nous avons traité par MQO les réseaux R6, R7 et R8 en faisant varier le paramètre IB de 90 à 30 . Il ressort de nos résultats numériques (tableau 9) que, pour un réseau donné, une fonction du type

$f(IB) = a\, b^{(100 - IB)/10}$, où a et b sont des constantes positives*, constitue un modèle qui prédit assez bien la valeur de T . La méthode MQO n'en demeure pas moins beaucoup plus rapide que la meilleure méthode optimale que nous ayons mis au point (option 3).

Si l'on revoit la définition de la méthode MQJM (section 2.6), on constate que son temps de calcul est directement proportionnel au temps d'évaluation des Q_j multiplié par le nombre d'arêtes éliminées du réseau initial R (nous négligeons ici l'étape (iv) qui ne constitue qu'un réajustement rapide de la solution trouvée). Grosso modo, le temps de calcul dépend donc linéairement du nombre d'arêtes éliminées, ce qui présente un avantage très net, par rapport à la méthode MQO , pour de petites valeurs de IB .

---

*Note: f(IB) est proportionnelle à $2^{-c * IB}$.

	MQJM				MQO			
problème	Z	T	NEL	Δ	Z	T	$f(IB)=ab^{(100-IB)/10}$	
(R6,90)	5398	1.22	3	0	5398	0.49	0.42	
(R6,80)	5408	2.01	4	+1	5407	0.65	0.62	
(R6,70)	5436	2.77	7	0	5436	0.85	0.89	a = 0.31
(R6,60)	5478	3.73	9	0	5478	0.68	1.24	b = $\sqrt{2}$
(R6,50)	5563	5.72	12	0	5563	1.84	1.75	
(R6,40)	5705	7.23	18	0	5705	2.41	2.48	
(R6,30)	LMIN > B							
(R7,90)	30465	1.15	3	0	30465	0.48	0.25	
(R7,80)	30489	1.94	5	0	30489	0.49	0.50	
(R7,70)	30664	3.38	8	+31	30633	1.00	1.00	a = $\frac{1}{4}$
(R7,60)	30924	4.48	12	-8	30932	1.53	2.00	b = 2
(R7,50)	31724	5.49	14	-14	31738	4.91	4.00	
(R7,40)	32928	6.51	17	0	32928	8.22	8.00	
(R7,30)	LMIN > B							
(R8,90)	99420	6.58	3	+8	99412	1.63	2	
(R8,80)	99614	10.39	7	+3	99611	1.79	4	
(R8,70)	100431	17.97	14	+70	100361	8.83	6	a = 1
(R8,60)	101959	21.61	16	-350	102309	15.87	16	b = 2
(R8,50)	104682	28.89	22	0	104682	16.09	32	
(R8,40)	132288	33.63	26			>300.	64	
(R8,30)	LMIN > B							

Remarque: Ici, Z et T correspondent respectivement à la valeur de la fonction économique et au temps de calcul pour la meilleure solution trouvée, qui n'est pas nécessairement optimale.

Δ = Z pour MQJM - Z pour MQO .

NEL = nombre d'arêtes éliminées.

Tableau 9

3.9 Conclusion

Notre travail visait tout d'abord à mettre au point un algorithme optimal qui accroisse les capacités de calcul des méthodes optimales existantes. Nous croyons y avoir réussi. En effet, comme l'indique un article de Sonia Strairs (1968) qui passe en revue la littérature sur le sujet, aucun chercheur ne rapporte de résultats numériques obtenus par une méthode optimale pour des réseaux de plus de 12 sommets. Depuis 1968, à notre connaissance, seuls Hoang (1971) et Boyce *et al.* (1973) ont présenté des algorithmes optimaux, accompagnés de résultats numériques. Et notre meilleur algorithme, l'option 3, peut avantageusement soutenir la comparaison. L'option 3 travaille beaucoup plus rapidement que l'algorithme de Hoang (par exemple, le réseau RH3 muni du budget B = 400 (IB ≅ 36) exige 13 secondes de l'algorithme de Hoang (option 1 avec FREQ = 3), tandis que l'option 3 prend moins de 1 seconde). L'option 3 permet de résoudre des problèmes qui restaient tout-à-fait inaccessibles à l'algorithme de Hoang (par exemple, pour le problème (R6,65) , T (Hoang) > 500 secondes alors que T (option 3) = 12.3 secondes). Dans le cas de l'algorithme de Boyce, nous ne pouvons actuellement faire des comparaisons que pour le réseau RB4 (le seul réseau décrit en détail dans la publication de Boyce *et al.*). Ces comparaisons restent un peu boîteuses, étant donné que Boyce *et al.* ont travaillé sur un ordinateur IBM 360/75, alors que nous disposions d'un CDC CYBER 74, mais nous les donnons à titre indicatif:

pour le problème (RB4,70), T (Boyce) ≅ 2 s. et T (option 3) = 0.8 s.

pour le problème (RB4,50), T (Boyce) ≅ 23 s. et T (option 3) = 0.7 s.

pour le problème (RB4,30), T (Boyce) > 1,000 s. et T (option 3) = 4.8 s.

Notre recherche nous a aussi permis de mettre en évidence le fait que la méthode MQJM (et aussi la méthode MQO , si IB n'est pas trop petit) constitue une alternative (aux méthodes optimales) très intéressante pour résoudre les problèmes de grande dimension. En effet, la méthode MQJM est peu coûteuse en temps de calcul et elle fournit une solution réalisable étonnamment proche de l'optimum (l'écart relatif Er ne dépassant guère 0.1%).

Références

Berge, C. (1967)
THEORIE DES GRAPHES ET SES APPLICATIONS
Dunod, Paris, 1967.

Boyce, D.E., Farhi, A. et Weischedel, R. (1973)
OPTIMAL NETWORK PROBLEM: A BRANCH-AND-BOUND ALGORITHM
Environment and Planning, vol. 5 (1973), pp. 519-533.

Dantzig, G.B. (1957)
DISCRETE VARIABLE EXTREMUM PROBLEMS
Oper. Res., vol. 5 (1957), pp. 266-277.

Dijkstra, E.W. (1959)
A NOTE ON TWO PROBLEMS IN CONNECTION WITH GRAPHS
Numerische Mathematik, 1, pp. 269-271.

Dionne, R. (1973)
ETUDE ET EXTENSION D'UN ALGORITHME DE MURCHLAND [...]
Publication # 122, département d'informatique,
Université de Montréal, mars 1973.

Floyd, R.W. (1962)
ALGORITHM 97 : SHORTEST PATH
Comm. of ACM 5 (1962), p. 345.

Hoang, H.H. (1971)
A COMPUTATIONAL APPROACH TO THE SELECTION OF AN OPTIMAL NETWORK
Document de travail # 10, département d'informatique,
Université de Montréal, août 1971.
(Note: publié aussi dans Management Science, vol. 19 no 5,
janvier 1973).

Lawler, E.L., and Wood, D.E. (1966)
BRANCH-AND-BOUND METHODS : A SURVEY
Oper. Res., vol. 14, no. 4, July -August 1966, pp. 699-719.

Roy, B. (1969)
PROCEDURE D'EXPLORATION PAR SEPARATION ET EVALUATION
R.I.R.O., 3e année, no v - 1, 1969, pp. 61-90.

Stairs, S. (1968)
SELECTING AN OPTIMAL TRAFFIC NETWORK
Journal of Transport Economics and Policy, vol. 2, pp. 218-231.